La Columna de Matemática Computacional

Sección a cargo de

Tomás Recio

El objetivo de esta columna es presentar de manera sucinta, en cada uno de los números de La Gaceta, *alguna cuestión matemática en la que los cálculos, en un sentido muy amplio, tengan un papel destacado. Para cumplir este objetivo el editor de la columna (sin otros méritos que su interés y sin otros recursos que su mejor voluntad) quisiera contar con la colaboración de los lectores, a los que anima a remitirle (a la dirección que se indica al pie de página*[1]*) los trabajos y sugerencias que consideren oportunos.*

En este número ...

... presentamos unas reflexiones magistrales y unos ejemplos clarificadores en torno a la Geometría Algorítmica o Computacional. Se trata de una joven línea de investigación matemática, a la que nuestro país empieza a contribuir de modo destacado. Los Encuentros Españoles de Geometría Computacional (EGC), con una década de existencia, son el lugar de puesta en común del trabajo de los pocos, pero entusiastas, grupos de investigación en este área. Tal vez uno de los más reputados por su proyección internacional sea el que lidera el profesor Ferrán Hurtado, de la Universitat Politècnica de Catalunya. La madurez y maestría de Ferrán para la divulgación son palpables en las páginas que siguen, en las que dibuja una instantánea muy atractiva de esa nueva forma de ver la Geometría.

[1]Tomás Recio. Departamento de Matemáticas. Facultad de Ciencias. Universidad de Cantabria. 39071 Santander. recio@matesco.unican.es

Geometría computacional: una instantánea

por

Ferrán Hurtado

INTRODUCCIÓN

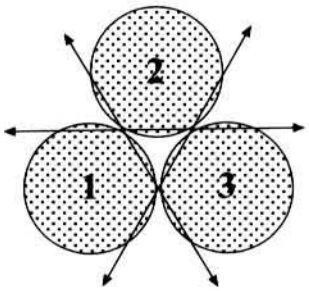
Sea $\mathcal{K} = \{K_1, \ldots, K_n\}$ una familia de conjuntos convexos compactos de $\mathbb{R}^d$, disjuntos dos a dos. Si existe una recta que corte a todos estos conjuntos, se dice que es una *transversal* de la familia. Hallar condiciones para la existencia de tal transversal es un problema clásico de la *geometría discreta*, y fue de hecho la primera gran generalización del Teorema de Helly. La noción de transversalidad se generaliza naturalmente a las k-variedades afines.

Una recta transversal induce una permutación de los conjuntos de la familia, dos si orientamos las rectas. Como puede verse en la Figura 1a, existen familias que admiten transversales que inducen distintos pares de permutaciones. Cada uno de estos pares recibe el nombre de *permutación geométrica*. Un problema típico de *geometría combinatoria* consiste en estudiar, cuando se consideran todas las familias $\mathcal{K}$ de cardinal n, cuál es el máximo número de permutaciones geométricas que se puede obtener. En el plano el problema se ha resuelto completamente ([KLZ, ES]) y la respuesta es $2n - 2$, mientras que está abierto en dimensión general. Si se trata de esferas, el orden asintótico es n^{d-1}, como se ha probado muy recientemente ([KLL, SMS]).

Desarrollar un algoritmo eficiente que permita, para cada familia específica dada, decidir si existe o no una recta transversal y, en caso afirmativo, encontrar una, es un problema que entra plenamente en el campo de la *geometría computacional*. En la versión más general lo que se quiere es especificar *todas* las variedades transversales de dimensión k dando una descripción finita de su realización en la correspondiente grassmaniana.

Sin duda que el lector ya se ha dado cuenta de que la tarea computacional que se quiere emprender es impensable sin una comprensión y resolución cabales de las condiciones discretas de existencia, y de que la complejidad de la solución depende de algún modo del problema combinatorio que hemos mencionado. La investigación en geometría computacional está siempre amalgamada con la investigación en los otros dos campos, y sería a menudo difícil (además de absurdo) decir qué partes de un trabajo corresponden a cada una de las tres geometrías.

La ejecución posterior del algoritmo correrá a cargo de un ordenador, previa transcripción a un programa, pero sin duda que el lector también ve la diferencia entre el desarrollo del algoritmo, actividad decididamente teórica en el caso que nos ocupa, y la actividad ulterior de programarlo.



(a)

51	23
25	46
12	~~92~~
6	~~184~~
3	368
1	736
	1173

(b)

FIGURA 1: (a) *Tres convexos con 6 permutaciones transversales.* (b) *Multiplicación de dos naturales*: $51 \times 23 = 1173$

Los problemas sobre cuerpos convexos, particularmente los relativos a la transversalidad, los recubrimientos, los empaquetamientos y las teselaciones, eran los temas centrales de la geometría discreta del segundo tercio de siglo. Helly, Fejes–Tóth, Hadwiger, Levi, Coxeter, Klee, Grünbaum, Yaglom y Boltjanski figuran entre los investigadores relevantes del área. En los años setenta (y aquí es obligatorio citar los nombres de P. Erdös y de D. Knuth) la matemática discreta empezó a experimentar una transformación y expansión muy aceleradas, por ser la disciplina parte crucial de los fundamentos teóricos de la informática, y desde entonces su desarrollo ha corrido en paralelo al incremento en disponibilidad y potencia de los ordenadores. Muchos temas de combinatoria y de computación han entrado a formar parte del área y, correlativamente, la geometría discreta se ha ido fusionando progresivamente con la geometría combinatoria y también con la geometría computacional, que es la subdisciplina más reciente y más centrada en los aspectos algorítmicos de los problemas geométricos. Como es muy frecuente que los fundamentos teóricos no estén disponibles y tengan que generarse, la geometría computacional, también llamada *geometría algorítmica*, es la fuente actual en que se generan la mayoría de problemas nuevos, ya que tiene una fuerte demanda de resultados desde sus campos de aplicación, casi todos relacionados con las nuevas tecnologías, como la informática gráfica, la robótica, el diseño y la fabricación asistidos por ordenador (CAD/CAM), el reconocimiento automático de formas, los sistemas de información geográfica, el diseño de VLSI y la visualización científica.

Como el problema de transversalidad con el que hemos empezado esta exposición requiere elementos avanzados, tanto en lo algorítmico como en cuanto a los fundamentos correspondientes, en este artículo describiremos la solución de otro problema, éste de naturaleza elemental, con la esperanza de que la interacción disciplinar a que aludíamos pueda ser entendida por una audiencia lo más amplia posible. Antes de ello, nos detendremos unos instantes para aclarar brevemente, en la próxima sección, a qué nos referimos cuando insistimos en que los algoritmos que queremos obtener han de ser *eficientes*.

COMPLEJIDAD ALGORÍTMICA

Cuando se dispone de más de un algoritmo para resolver un problema, es natural preguntarse cuál es el mejor, pero la respuesta puede variar según qué criterio de calidad se use, como pueden ser la elegancia, la simplicidad, la comprensibilidad, la robustez... Un bien conocido ejemplo lo da el siguiente algoritmo para multiplicar dos naturales (Figura 1b): empecemos una columna con uno de ellos y anotemos en ella los resultados de dividirlo sucesivamente por dos, tomando la parte entera, hasta alcanzar la unidad, mientras que en otra columna vamos duplicando el otro natural el mismo número de veces; sumamos los números de la segunda columna que corresponden a números impares en la primera, el resultado es el producto que buscábamos. Obsérvese que para ejecutar este algoritmo basta con saberse la tabla del 2. Dejamos al lector la prueba de corrección y los considerandos sobre si este algoritmo es mejor o peor que el que se enseña en nuestras escuelas.

Cuando el algoritmo ha de ser utilizado por una máquina, un criterio fundamental es la rapidez de ejecución. En determinadas disciplinas y situaciones prácticas es corriente implementar los algoritmos (es decir, programarlos), y contrastar empíricamente su comportamiento en una misma máquina contra diversas muestras de datos. Pero en matemática computacional la aspiración es poder *demostrar* de manera estrictamente formal que cierto algoritmo es superior a los restantes.

Para ello se utiliza un modelo teórico de máquina en el que ciertos procesos tienen asignado un coste (=tiempo) unidad. El modelo más usado en geometría computacional es el *RAM real*, en el que se acepta que un número real se puede almacenar en una unidad de memoria, y las operaciones siguientes se consideran *primitivas*, pudiéndose realizar con coste unitario:

(1) acceso a la memoria;
(2) operaciones aritméticas básicas ($+$, $-$, $\times$, $/$);
(3) comparación de dos números reales ($<$, $\leq$, $=$, $\neq$, $\geq$, $>$);
(4) (ocasionalmente) raíces enésimas, funciones trigonométricas, la exponencial y el logaritmo.

El comportamiento de un algoritmo depende del conjunto específico de datos sobre el cual se ejecuta cada vez. El *análisis en el peor caso*, el único que describiremos aquí, consiste en calcular el coste total de ejecutar el algoritmo (=número de operaciones primitivas) *en la situación más desfavorable.* La cantidad de datos recibidos por el algoritmo se mide en función de algún parámetro n, y el coste será una función $T(n)$. Como los algoritmos se diseñan para grandes valores de n, no se calcula $T(n)$ "exactamente", sino en orden asintótico de magnitud, ignorando los coeficientes constantes. Escribimos $T(n) \in O(f(n))$ para acotar superiormente el tiempo de ejecución $T(n)$ de un algoritmo. Más formalmente:

$$T(n) \in O(f(n)) \iff \exists n_0 \in \mathbf{N}\,,\ \exists \alpha \in \mathbf{R}^+,\ \text{tales que}\ T(n) \leq \alpha \cdot f(n)\ \ \forall n \geq n_0.$$

Así, si hemos de calcular la distancia entre dos puntos del plano, tardaremos $O(1)$ (¡o sea, tiempo constante!), y si hemos de leer una matriz $2n \times 300n$ tardaremos $O(n^2)$. Con este modelo, la comparación de la eficiencia de dos algoritmos ha quedado, pues, reducida a la comparación asintótica de las funciones de coste correspondientes.

Vale la pena detenerse a valorar la enorme importancia de que se tenga un coste u otro. Por ejemplo, hay una diferencia abismal entre disponer, para resolver cierto problema, de un algoritmo de orden $O(n \log n)$ o bien de uno $O(n^2)$. Si la unidad de medida es el milisegundo y hay 10000 datos, el primero tardaría del orden de 10000×4 milisegundos $= 40$ segundos, mientras que el segundo tardaría unos 10000×10000 milisegundos, ¡casi 28 horas!; haga el lector las cuentas si lo que se tiene es un algoritmo $O(n^4)$ o $O(2^n)$. De hecho, muchos avances espectaculares en la rapidez de los ordenadores que los profanos atribuyen siempre a las innovaciones en el *hardware* son en realidad mejoras en los algoritmos (que se traducen en el *software*), las cuales, como acabamos de ver, incrementan mucho más drásticamente la rapidez de ejecución.

La complejidad de un problema $\mathcal{P}$ se define como el mínimo de los costes de los algoritmos que lo resuelven. Por lo tanto, el coste de cada algoritmo que solucione $\mathcal{P}$ es una cota superior de la complejidad de $\mathcal{P}$. Si se dispone de una cota inferior, es decir un valor por debajo del cual no puedan existir algoritmos que resuelvan $\mathcal{P}$, y de un algoritmo que tenga como coste esa cota, se dice que tal algoritmo es *óptimo*. Ese es el estado idílico de un problema, aunque desgraciadamente en muchos casos no se sabe la cota inferior, o no se conocen algoritmos, o las mejores cotas disponibles están separadas.

La obtención de cotas inferiores para la complejidad de problemas es un área muy interesante, que no podemos abordar en este artículo, en la que la geometría computacional se solapa con la topología y con la geometría semialgebraica, ya que varios resultados relacionan la dificultad de obtener una solución con la estructura topológica (por ejemplo el número de componentes conexas o los números de Betti) del subconjunto semialgebraico cuyos puntos corresponden a las soluciones del problema, dentro de un espacio de fases adecuado. El lector interesado puede consultar, por ejemplo, [Re, Sa].

UN PROBLEMA SENCILLO DE GEOMETRÍA COMPUTACIONAL

Empezaremos con algunas definiciones. Sea K un conjunto convexo compacto del plano; se dice que una recta r es *de soporte* de K en un punto P si $P \in (K \cap r)$ y K yace enteramente en uno de los semiplanos cerrados definidos por r. Un *polígono simple de n lados* es la curva cerrada de Jordan definida por la concatenación de n segmentos que comparten extremos con sus adyacentes, y se especifica dando las coordenadas de sus vértices en el orden circular en que éstos aparecen antihorariamente en la frontera.

Disponer de un rectángulo que contenga un objeto es útil en varias situaciones, como en el tratamiento de imágenes, en problemas de empaquetamiento o en la planificación de trayectorias (si dos cajas englobantes no colisionan tampoco lo hacen los objetos contenidos). Naturalmente, el problema intere-

sante consiste en optimizar dicho rectángulo. Aquí resolveremos una versión muy específica.

Problema 1. *Hallar un algoritmo que permita calcular, para cada polígono convexo Q de n lados que reciba, el rectángulo de área mínima que contiene a Q ("rectángulo contenedor mínimo").*

Decimos que un rectángulo que contenga a Q es óptimo en una dirección dada, si dos de sus lados son paralelos a esa dirección y los cuatro lados extendidos son rectas de soporte de Q. Obviamente el rectángulo contenedor mínimo de Q es óptimo en alguna dirección.

Si una recta r es de soporte de Q a lo largo de todo un lado, consideramos punto *propio* de soporte al que se obtiene girando r infinitesimalmente en el sentido antihorario. A cada rectángulo óptimo R le asociamos el conjunto formado por los cuatro vértices del polígono en que los lados de R soportan propiamente Q (son cuatro en general, podrían ser tres o incluso dos). Decimos que dos rectángulos óptimos son de la misma clase si tienen el mismo conjunto de puntos propios de soporte.

Es obvio que para los de una misma clase podemos hallar en tiempo constante el rango de variación angular y averiguar (con herramientas de cálculo elemental) qué representante tiene área mínima . Esto sugiere que podemos resolver el Problema 1 considerando sucesivamente todas las $\binom{n}{4}$ cuaternas de vértices y para cada una, en tiempo constante, ver si admite alguna cuaterna de rectas de soporte que formen rectángulo y, en caso afirmativo, seleccionar el mejor rectángulo de la clase, comparándolo con la mejor solución obtenida hasta ese momento. Puesto que $\binom{n}{4} \in O(n^4)$, queda demostrada la proposición siguiente.

Proposición 1. *El rectángulo de área mínima que contiene a un polígono convexo de n lados puede calcularse en tiempo $O(n^4)$.*

El algoritmo anterior, que en la jerga del oficio se llama *de fuerza bruta*, es aceptable cuando el polígono tiene una docena de puntos, pero si tiene 12000 es absolutamente inútil. En geometría computacional hay que imaginarse siempre que n es un número muy grande, no sólo porque para polígonos de juguete no hacen falta sofisticaciones algorítmicas, sino porque son esos los números que se barajan en las aplicaciones: una imagen de animación tridimensional de una película contiene cientos de miles de polígonos, y una red geodésica contiene millones de puntos. El polígono con el que estamos tratando seguramente parecería a la vista un convexo suave de frontera curvilínea.

Parece muy difícil que cada una de las $\binom{n}{4}$ cuaternas de vértices sean capaces de dar soporte a un rectángulo. Ello sugiere el problema siguiente, de naturaleza combinatoria.

Problema 2. *¿Cuál es el número de cuaternas distintas en que un polígono convexo Q de n lados puede ser tocado por los cuatro lados de un rectángulo contenedor?*

Obsérvese que la respuesta puede ser la misma para todos los n-gonos convexos, o depender del particular Q; de ser así, habremos de preocuparnos del peor caso posible. En cualquier caso, si conseguimos demostrar, por ejemplo, que el número de cuaternas posibles es $O(n^2)$, tendremos esperanza de obtener un algoritmo de eficiencia mejor que la que da la Proposición 1. Sin embargo, aunque lo consiguiéramos, todavía quedaría el problema de detectarlas, pues hemos de evitar examinarlas todas para hacer el descarte.

Un teorema de geometría discreta, en el sentido más clásico, es lo que permite romper la dificultad. Se trata, en este caso, de una condición necesaria que ha de cumplir el rectángulo contenedor mínimo.

Teorema 1. *El rectángulo de área mínima que contiene a un polígono convexo Q tiene uno de sus lados contenido en la recta determinada por extensión de un lado de Q.*

Dejamos al lector la demostración, que apareció en [FS]. Ahora disponemos de un algoritmo mucho mejor, pues basta considerar por turno cada uno de los n lados de Q, extenderlo a una recta y hallar en tiempo $O(n)$ la paralela y las dos perpendiculares que completan el rectángulo óptimo asociado a la dirección. Puesto que esta etapa tiene coste lineal y se repite n veces, tenemos:

Proposición 2. *El rectángulo de área mínima que contiene a un polígono convexo de n lados puede calcularse en tiempo $O(n^2)$.*

La solución anterior es la que aparecía en [FS]. Pero además ahora nos damos cuenta de que era fácil resolver el Problema 2, ya que cada clase de equivalencia de cuaternas contiene exactamente un representante en que una de las rectas sea extensión de un lado del polígono: basta girar horariamente un representante que no esté en tal situación. Por lo tanto tenemos:

Teorema 2. *El número de cuaternas distintas en que un polígono convexo Q de n lados puede ser tocado por los cuatro lados de un rectángulo contenedor es $O(n)$, y es exactamente n si Q no tiene pares de lados paralelos ni perpendiculares.*

Como ya se imagina el lector, este resultado abre el interrogante de si es posible hallar un algoritmo más eficiente que el de la Proposición 2. Pues sí, es posible, como se demostró en [To]:

Teorema 3. *El rectángulo de área mínima que contiene a un polígono convexo de n lados puede calcularse en tiempo $O(n)$.*

Demostración. Tomemos un lado arbitrario de Q, extendámoslo a una recta, y completemos en tiempo $O(n)$ el rectángulo óptimo asociado a esa dirección. Cada una de las cuatro rectas tiene cierto punto de apoyo P, y determina un ángulo posible de giro antihorario en torno a P hasta que se apoya en el lado siguiente (Figura 2). Tomemos el menor de esos cuatro ángulos y hagamos girar solidariamente las cuatro rectas. De ese modo se determina en tiempo constante un nuevo rectángulo contenedor cuya área calcularemos. Iteremos

esta operación, y observemos que en cada paso al menos uno de los lados del cuadrilátero avanza un vértice sobre la frontera. Tras girar $\pi/2$, se han avanzado n vértices (colectivamente) y se han dado $O(n)$ pasos, volviendo a la situación inicial tras haber pasado por cada uno de los rectángulos apoyados en un lado de Q, lo que, en virtud del Teorema 1, prueba el resultado. □

El método empleado en este algoritmo es el llamado *de los calibres giratorios*, porque simula el giro de una herramienta amordazando el polígono, y tiene muchas otras aplicaciones que se describen en [To]. Por otra parte, la complejidad del problema que nos ocupa tiene una cota inferior lineal que es trivial, pues cualquier algoritmo que resuelva el Problema 1 tiene que leer las coordenadas de los vértices del polígono, lo que ya requiere tiempo lineal en n. Por lo tanto el algoritmo del Teorema 3 es óptimo: ¡por fin podemos descansar!

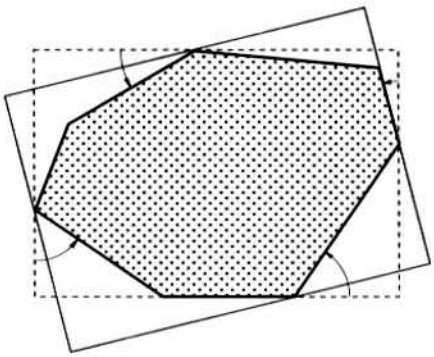


FIGURA 2. *Cálculo del menor rectángulo contenedor por calibres giratorios*

EXTENSIONES DEL PROBLEMA

El Problema 1 se ha generalizado en direcciones muy diversas. Por ejemplo, puede considerarse perímetro en vez de área, círculos o triángulos equiláteros en vez de rectángulos, o incluso figuras contenedoras semejantes a una dada y que se conoce como dato al propio tiempo que el polígono que se quiere circunscribir [OAMB, We]. Otra dirección natural es el problema correspondiente en tres dimensiones: encontrar el ortoedro de volumen mínimo que contiene a un politopo (véase [OR]).

Otra línea de investigación es la siguiente: en vez de tener un solo polígono, supongamos que tenemos varios, y que queremos desplazarlos superponiéndolos de forma que se pueda obtener el menor rectángulo que los contenga simultáneamente a todos. La misma cuestión puede plantearse en términos de empaquetamiento, es decir, sin permitir solapamientos. Estos problemas se resuelven en [AH], de donde reproducimos un resultado a modo de ejemplo.

Teorema. *Sean P y Q dos polígonos convexos del plano con un total de n vértices entre ambos. En tiempo $O(n \log n)$ puede obtenerse el mínimo rectángulo que puede contener simultáneamente copias isométricas solapadas de P y de Q.*

Nos limitaremos aquí a esbozar la solución. Consideremos un plano auxiliar F, dotado de un sistema de coordenadas cartesianas rectangulares, y asociemos al rectángulo de base x y altura y el punto (x, y) de F. De este modo, tenemos una biyección entre todos los rectángulos posibles y el primer cuadrante de F, sin los semiejes.

Dado un polígono Q de n lados consideremos todos los rectángulos que contienen a Q tocándole en sus cuatro lados. El lugar geométrico de los puntos que les corresponden en F es una curva cerrada $\mathcal{C}(Q)$, en general no simple, formada por la concatenación de n arcos de elipse.

Se dice que un punto (o un vector) (a_1, b_1) *domina* a otro (a_2, b_2) cuando $a_1 \geq a_2$ y $b_1 \geq b_2$. La dominación induce un orden parcial $\mathcal{D}$ en el plano. Un punto $(x, y) \in F$ corresponderá a un rectángulo capaz de contener a Q si y sólo si domina a un punto de $\mathcal{C}(Q)$, por lo que el lugar de dichos puntos es una región $R(Q)$ como la que aparece sombreada en la Figura 3a, donde la línea zigzagueante son los puntos de $\mathcal{C}(Q)$ minimales con respecto al orden $\mathcal{D}$. Los rectángulos que pueden contener copias isométricas de P y de Q se corresponderán con los puntos de la región $R(P) \cap R(Q)$.

Finalmente, los puntos de F que corresponden a rectángulos de área k forman la rama positiva de la hipérbola $xy = k$. Al variar k, consideremos el menor valor que da intersección no vacía con la región $R(P) \cap R(Q)$: el punto de contacto nos da las coordenadas del rectángulo buscado (Figura 3b). En [AH] se demuestra que la construcción de la región $R(P) \cap R(Q)$ y del punto de contacto pueden hacerse dentro de la complejidad citada en el Teorema 3.

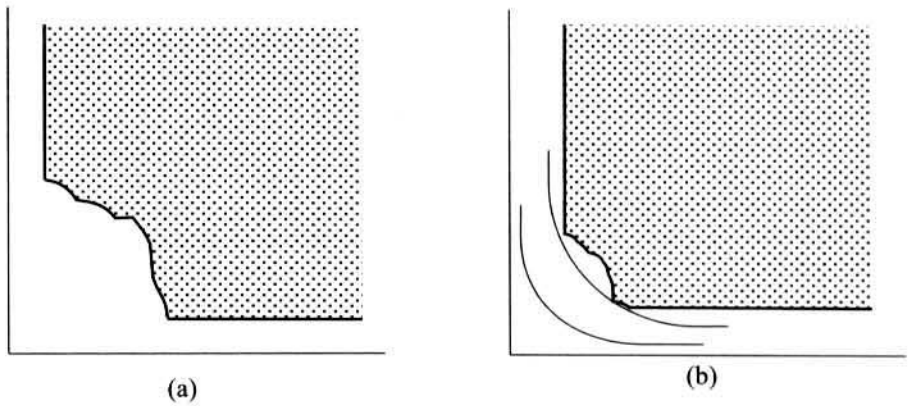


FIGURA 3. (a) *Puntos asociados a rectángulos contenedores.* (b) *Punto de contacto para la circunscripción simultánea óptima.*

LECTURAS ADICIONALES

Esperamos que esta breve exposición haya dejado al lector con ganas de saber más del tema. Si es así, de entre las numerosas referencias interesantes existentes, sugerimos [BY] para una introducción a la geometría computa-

cional, [PA] para los temas más centrados en la geometría discreto-combinatoria, y [CLR] para acceder al mundo teórico del análisis y diseño de algoritmos.

Bibliografía

[AH] ALT H., HURTADO F.: *Optimal rectangles simultaneously enclosing several objects with and without overlapping.* Manuscrito en preparación.

[BY] BOISSONNAT J-D., YVINEC M.: *Géométrie Algorithmique*, Ediscience International, 1995. Existe una traducción al inglés: *Algorithmic Geometry*, Cambridge University Press, 1997.

[CLR] CORMEN T., LEISERSON C., RIVEST R.: *Introduction to Algorithms.* Ed. MIT Press, 1990.

[ES] EDELSBRUNNER H., SHARIR M.: *The maximum number of ways to stab n convex non-intersecting sets in the plane is* $2n - 2$. Discrete and Computational Geometry, 5(1):35-42, 1990.

[FS] FREEMAN H., SHAPIRA R.: *Determining the minimum area encasing rectangle for an arbitrary closed curve.* Communications of the ACM, 18:409-413, 1975.

[KLL] KATCHALSKI M., LEWIS P., LIU L.: *The different ways of stabbing disjoint convex sets.* Discrete and Computational Geometry, 7:197-206, 1992.

[KLZ] KATCHALSKI M., LEWIS P., ZACKS J.: *Geometric permutations for convex sets.* Discrete Mathematics, 54:271-284, 1992.

[OAMB] O'ROURKE J., AGGARWAL A., MADDILA S., BALDWIN M.: *An optimal algorithm for finding minimal enclosing triangles.* J. Algorithms, 7:258-269, 1986.

[OR] O'ROURKE J.: *Finding minimal enclosing boxes.* Intern. J. Comput. Inform. Sci., 14:183-199, 1985.

[PA] PACH J., AGARWAL P.: *Combinatorial Geometry.* Wiley & Sons, 1995.

[Re] RECIO T.: *Searching for lower bounds in computational geometry. A survey on methods.* Actas de los VI Encuentros de Geometría Computacional, 7-21, Barcelona, 1995.

[Sa] SACRISTÁN V.: *Cotas inferiores para problemas geométricos.* IR del Dept. de Mat. Aplic. II, Univ. Pol. de Catalunya. Manuscrito en preparación.

[SMS] SMORODINSKY S., MITCHELL J., SHARIR M.: *Sharp bounds on geometric permutations of pairwise disjoint balls in* $\mathbb{R}^d$. Proc. of the 15th ACM Symp. on Comp. Geom., 400-406, 1999.

[T] TOUSSAINT G.: *Solving geometric problems with the "rotating callipers".* Proc. of IEEE MELECON, 1983.

[We] WELZL E.: *Smallest enclosing disks (balls and ellipsoids).* En *New Results and New Trends in Computer Science*, H. Maurer, ed., Lecture Notes Comput. Sci. 555:359–370, Springer-Verlag, 1991.

Ferran Hurtado,
Departament de Matemàtica Aplicada II,
Universitat Politècnica de Catalunya,
C/ Pau Gargallo, 5, 08028 Barcelona.
e-mail: hurtado@ma2.upc.es